Open in app

Follow 593K Followers

You have **2** free member-only stories left this month. Upgrade for unlimited access.

Advanced sports visualization with Python, Matplotlib and Seaborn

Detailed tutorial on how to derive valuable insights from the most competitive games

Tuan Nguyen Doan Nov 9, 2018 · 9 min read ★

Photo by Lukas Blazek on Unsplash

“The greatest value of a picture is when it forces us to notice what we never expected to see.”

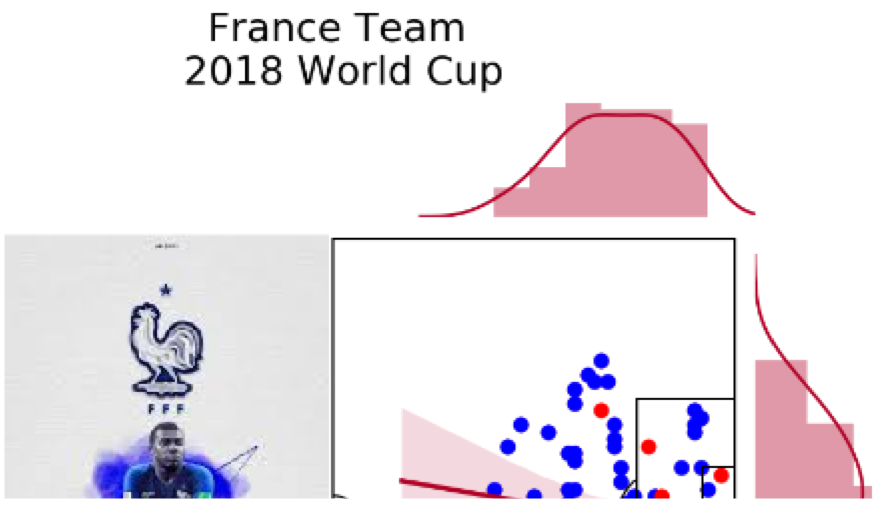
You have seen it all in the news: media outlets raving about the Coming of Sports Analytics age, commercial ads picturing Big Data tools like a pocket calculator, punditry using multi-dimensional cameras to collect details of every single perspiration of sport players ***enter your childhood hero names: L.Messi, LeBron James, Tom Brady, etc*** and enhance their performance by 10x.

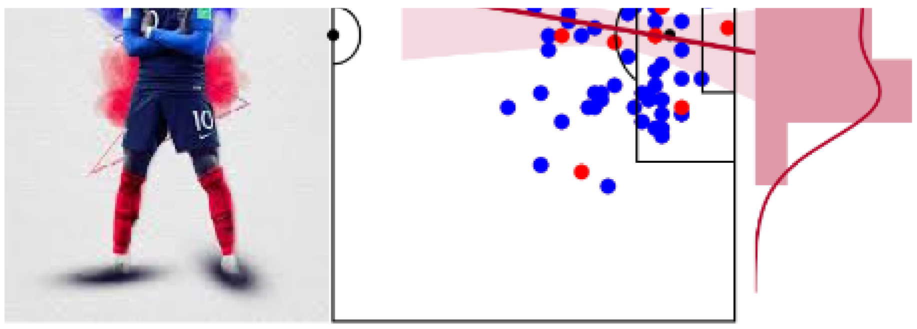
While obviously we are not quite there yet, Sports Analytics have come a long way in helping teams with respect to decision making, which have traditionally been made by “gut” feeling or adherence to past traditions, as to which players to draft, trade, develop, coach and which system to play.

In this blog posts, I will not develop an algorithm to predict how likely (unlikely) it is that Tottenham will get to the top 4 of the Premier League, or how to build a perfect Jets team for 2018. Instead, I will introduce some visualization techniques that will help us prepare eye-catching and insightful graphs to woo your fellow sport enthusiasts.

Note:

- Even though my example remains exclusively in Football and the FIFA World Cup, the techniques are applicable to any sports and tournaments.
- You can find all the source code for this tutorial here.





Details of all shots France team attempted across their 7-match World Cup campaign. You will be able to create this visualization by the end of this tutorial. Original image of Mbappe is by adi-149 via DevianArt

1. Getting the data:

Bad news: this is usually the hardest part

While you may have no trouble collecting summary statistics of a sport events (goals in a football match, throws in a basketball game, etc.), it is usually a lot harder find a detailed, play-by-play dataset of a particular football game or a tennis match. Tracking players on the field, especially for high-tempo, high intensity sports such as football or basketball is a formidable task, albeit a very lucrative one. Companies such as SportVu or Opta Sports monetize by selling these highly sought-after information to teams, sport consultancies or research centers.

> *"For every match we have three guys using[…] a live video feed on a pitch graphic: one guy watches the home team, another does the away team and the third man is essentially a data checker"*

— Simon Banoub, Opta's director of Marketing

Here comes our unsung hero

Recently, Statsbomb announced the public release of the play-by-play, second-accurate datasets of every single game across the three recent football leagues: the National Women Soccer League (US), the FA Women's Super League (England), and the 2018 FIFA World Cup. You can get access to their datasets for free here.

The datasets are all in json format, so you will need to parse the raw dataset to a relational format that can be easily retrieved and manipulated.

```python
import json
from pandas.io.json import json_normalize

with open('./Germany_Korea.json') as data_file:
    data = json.load(data_file)
df = json_normalize(data, sep = "_")
```

visual6.py hosted with ♡ by GitHub view raw

json_normalize() is a very convenient tools as it can automatically "normalize" json structure into a flat relational structure.

In this tutorial, we will focus exclusively in the 2018 FIFA World Cup. I encourage you to read the documentation of the dataset before diving in with the analysis. After all, understanding your data is paramount.

2. Drawing a football pitch:

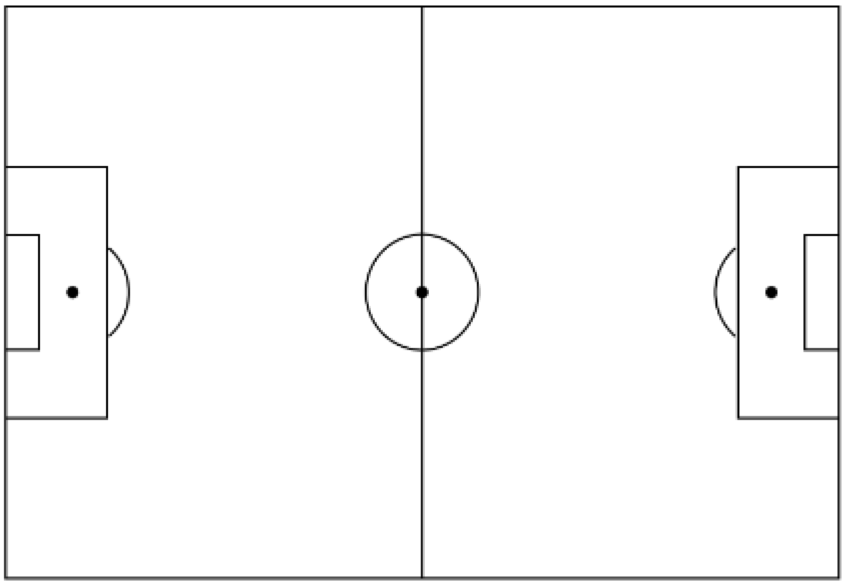
At first, let us use Matplotlib to draw a simple football pitch.

```python
def draw_pitch(ax):
    # focus on only half of the pitch
    #Pitch Outline & Centre Line
    Pitch = Rectangle([0,0], width = 120, height = 80, fill = False)
    #Left, Right Penalty Area and midline
    LeftPenalty = Rectangle([0,22.3], width = 14.6, height = 35.3, fill = False)
    RightPenalty = Rectangle([105.4,22.3], width = 14.6, height = 35.3, fill = False)
    midline = ConnectionPatch([60,0], [60,80], "data", "data")

    #Left, Right 6-yard Box
    LeftSixYard = Rectangle([0,32], width = 4.9, height = 16, fill = False)
    RightSixYard = Rectangle([115.1,32], width = 4.9, height = 16, fill = False)


    #Prepare Circles
    centreCircle = plt.Circle((60,40),8.1,color="black", fill = False)
    centreSpot = plt.Circle((60,40),0.71,color="black")
    #Penalty spots and Arcs around penalty boxes
    leftPenSpot = plt.Circle((9.7,40),0.71,color="black")
    rightPenSpot = plt.Circle((110.3,40),0.71,color="black")
    leftArc = Arc((9.7,40),height=16.2,width=16.2,angle=0,theta1=310,theta2=50,color="black")
    rightArc = Arc((110.3,40),height=16.2,width=16.2,angle=0,theta1=130,theta2=230,color="black

    element = [Pitch, LeftPenalty, RightPenalty, midline, LeftSixYard, RightSixYard, centreCirc
               centreSpot, rightPenSpot, leftPenSpot, leftArc, rightArc]
    for i in element:
        ax.add_patch(i)
```

That seems a lot, but let's unpack the **draw_pitch()** function line by line. The function takes in an ax argument, which is the output of the **add_subplot()** function in Matplotlib. It then adds several objects with pre-defined dimension to recreate an image of a football pitch, including the center circle, the penalty areas, the 6-yard boxes, and the arcs in the pitch. Once we have defined this function, we call in together with standard Matplotlib figure function as follows:

```
fig=plt.figure() #set up the figures
fig.set_size_inches(7, 5)
ax=fig.add_subplot(1,1,1)
draw_pitch(ax) #overlay our different objects on the pitch
plt.ylim(-2, 82)
plt.xlim(-2, 122)
plt.axis('off')
plt.show()
```

visual2.py hosted with ♡ by GitHub view raw


A nice football pitch constructed with no more than rectangles, circles and arcs

3. Level up your visualization with a Pass Map and a Heat Map:

What is the most shocking moment of the 2018 FIFA World Cup?

Image by Дмитрий Садовников via Wikipedia

There are many contenders but few would beat the moment when then-defending champion Germany crashed out of the World Cup after defeat by South Korea. There were pains, there were tears, and surely, there were criticism. The aftermath of the sobering defeat saw the departure of German's number 10, Mesut Özil, who bore almost the entire blame, both for his performances and his meeting with Turkish president Erdoğan right before the World Cup. Just listen to what Bayern Munich's president Uli Hoeness had to say about Özil:

> *"He had been playing s*** for years[…] And now he and his s*** performance hide beyond this picture."*

But, was he even that bad?

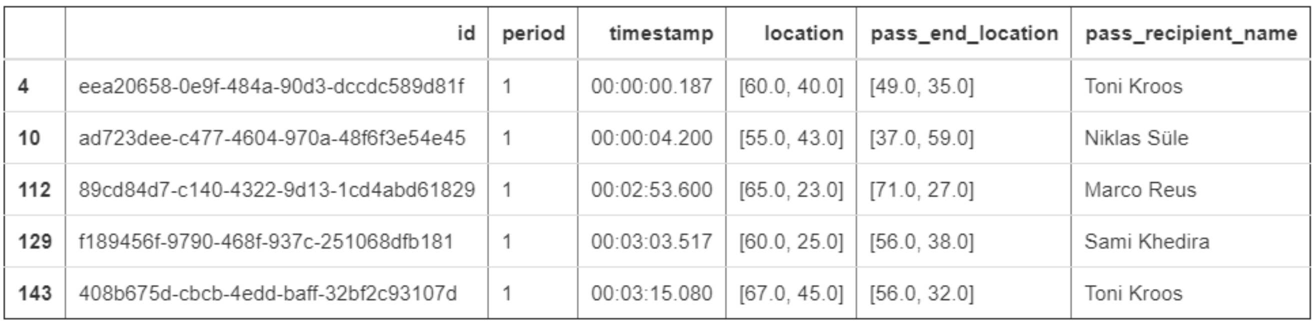
Let's look at his performance during the widely criticized match against South Korea. I want to plot a heat map and a pass map to capture his performances during the 90 minutes and to evaluate the influence (whether positive or negative) he exerted on the German's offensive side.

Let's start with a Pass Map

We load the json file and do some basic data cleaning in Panda to get a dataset that only contains Passing Events by Mesut Özil.

```python
# loading the json file
ozil_pass = df[(df['type_name'] == "Pass") & (df['player_name']=='Mesut Özil')] # get passing ir
pass_column = [i for i in df.columns if i.startswith("pass")]
ozil_pass = ozil_pass[["id", "period", "timestamp", "location", "pass_end_location", "pass_recip
```

visual3.py hosted with ♡ by GitHub view raw



	id	period	timestamp	location	pass_end_location	pass_recipient_name
4	eea20658-0e9f-484a-90d3-dccdc589d81f	1	00:00:00.187	[60.0, 40.0]	[49.0, 35.0]	Toni Kroos
10	ad723dee-c477-4604-970a-48f6f3e54e45	1	00:00:04.200	[55.0, 43.0]	[37.0, 59.0]	Niklas Süle
112	89cd84d7-c140-4322-9d13-1cd4abd61829	1	00:02:53.600	[65.0, 23.0]	[71.0, 27.0]	Marco Reus
129	f189456f-9790-468f-937c-251068dfb181	1	00:03:03.517	[60.0, 25.0]	[56.0, 38.0]	Sami Khedira
143	408b675d-cbcb-4edd-baff-32bf2c93107d	1	00:03:15.080	[67.0, 45.0]	[56.0, 32.0]	Toni Kroos

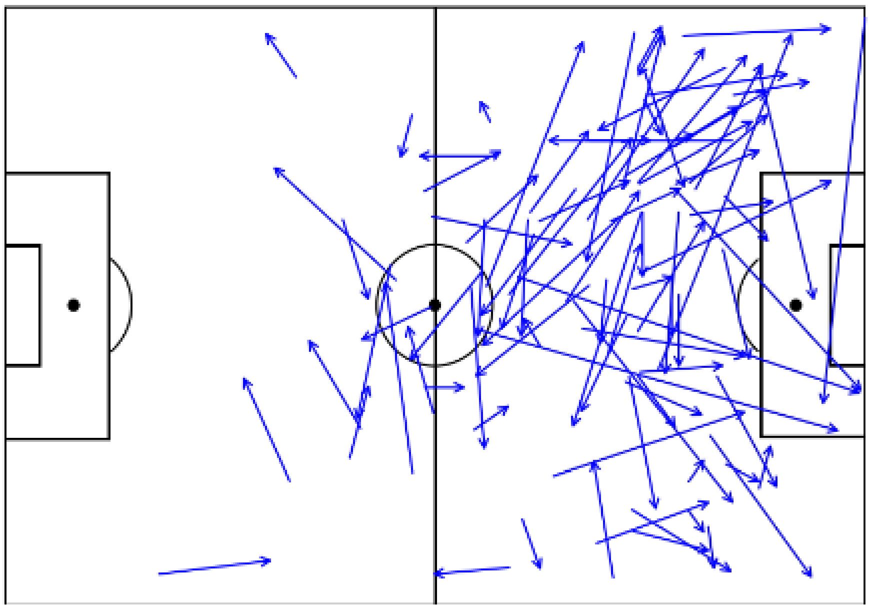
What our condensed dataset looks like. You can extract more information: pass_complete_status, etc.

This dataset is extremely meaningful, i.e., one can find out that Ozil attempted as many as 95 passes with up to 7 incisive ones in the match, which was pretty impressive for an attacking midfielder, or he relayed the ball the most to Toni Kroos (19 times) and Marco Reus (18 times) during the game. For the purpose of the pass map, we only care about the starting and ending location of a pass

The code below allows us to overlay the passes as arrows onto our pitch

```python
for i in range(len(ozil_pass)):
    # annotate draw an arrow from a current position to pass_end_location
    ax.annotate("", xy = (ozil_pass.iloc[i]['pass_end_location'][0], ozil_pass.iloc[i]['pass_enc
                xytext = (ozil_pass.iloc[i]['location'][0], ozil_pass.iloc[i]['location'][1]), te
                arrowprops=dict(arrowstyle="->",connectionstyle="arc3", color = "blue"),)
```

visual4.py hosted with ♡ by GitHub view raw



Looks pretty good but we can do even better. I will come back to how a little tweak can make this plot a lot more informative

Tracking active zone with a heat map

Football heatmaps are used by in-club and media analysts to illustrate the area within which a player has been present. They are effectively a smoothed out scatter plot of player locations and could be a good indicator of how effective a player is at different parts of the field. While there may be some debate as to how much they are useful (they don’t tell you if actions/movement are a good or bad thing!), they can often be very aesthetically pleasing and engaging, hence their popularity.

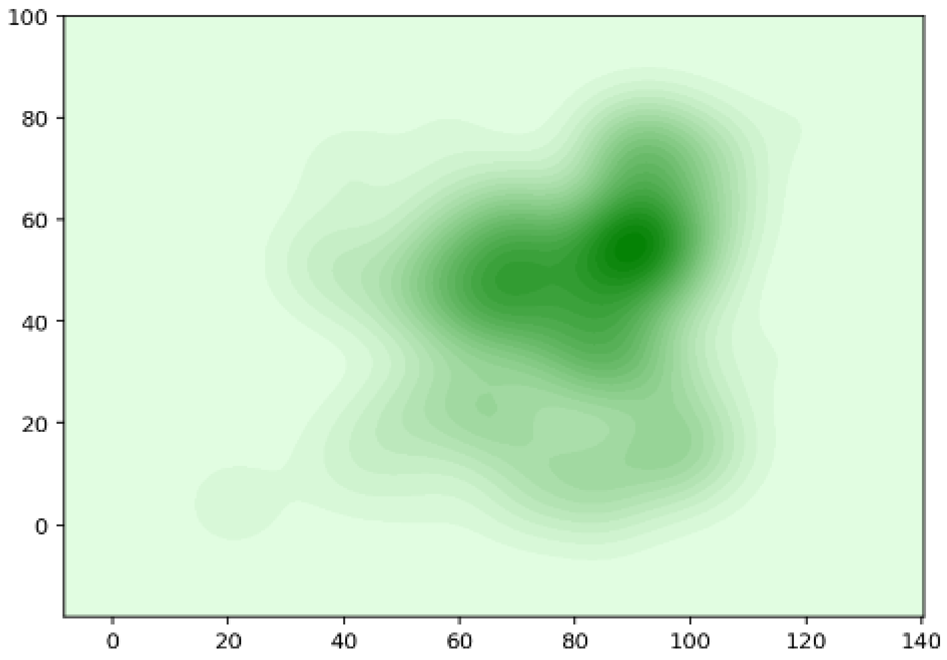
Now if we go back to Mesut Özil, one of the main criticisms he faces is the low amount of field coverage as we rarely see him launching in tackles or fighting for possession, thus, “low level of work rate” as they say.

But is that really the case?

Let’s plot a heat map using Seaborn on top of Matplotlib to visualize Mesut Ozil’s involvement during 90-minute of the Germany-Korea match. The syntax of the code is incredibly simple. We use a kdeplot, which will draw a kernel density estimate of the scattering points of Özil’s locations.

```python
fig, ax = plt.subplots()
fig.set_size_inches(7, 5)

x_coord = [i[0] for i in ozil_action["location"]]
y_coord = [i[1] for i in ozil_action["location"]]

#shades: give us the heat map we desire
# n_levels: draw more lines, the larger n, the more blurry it looks
sns.kdeplot(x_coord, y_coord, shade = "True", color = "green", n_levels = 30)
plt.show()
```

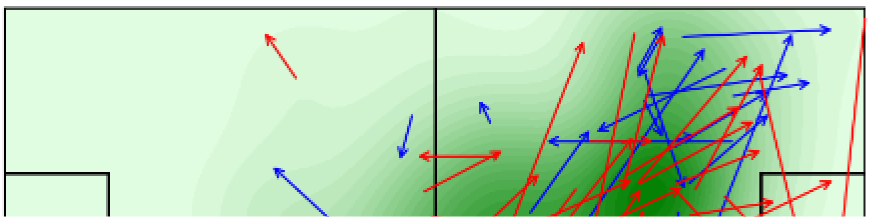
visual5.py hosted with ♡ by GitHub view raw

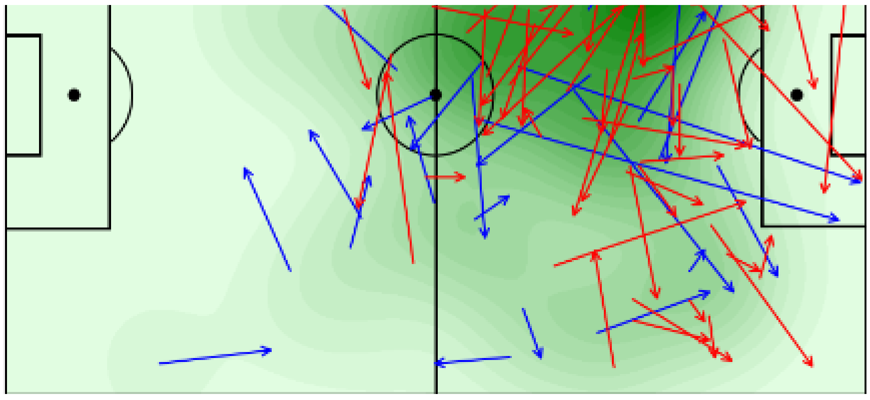


Wow!!! That looks very… anti-climatic. After all, what is the graph trying to tell you? I see some coordinates, and clearly these contour-looking plots does seem to indicate that Özil is more active in the area with darker color.

Can we do any better than that?

Yes, the answer is that we can combine (1) the pitch, (2) the pass map and (3) the heat map in order to have a more comprehensive views of Ozil's performance during the game



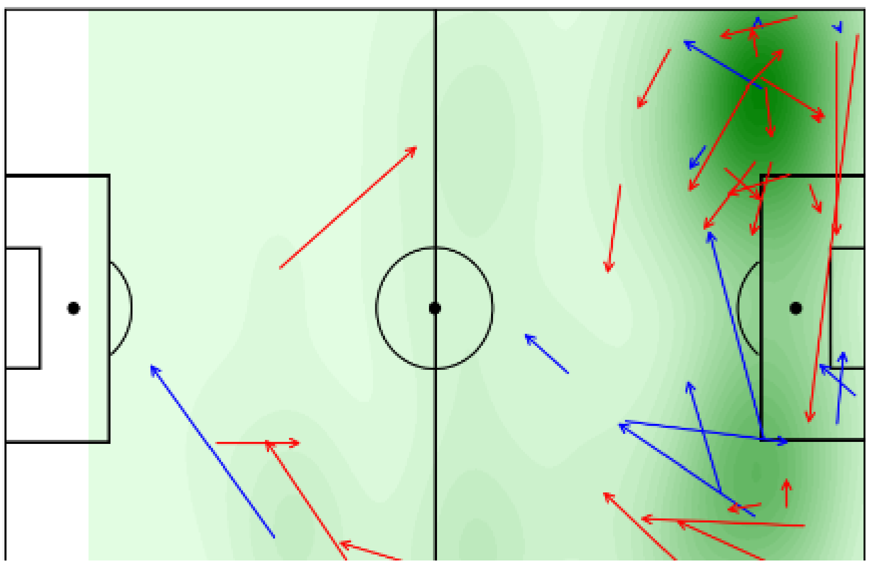


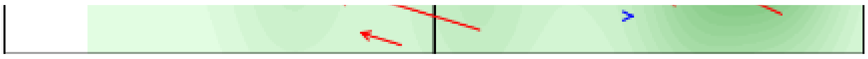
Notice that I also color the passes differently, as the blue arrows indicate passes made in the first half, and the red arrows second half

Now we can see a more comprehensive picture of Mesut Özil's performance during the game. A couple of observations right off the bat:

- He covered almost exclusively the opponent's half, so criticisms against his lack of defensive mindset are not completely unfounded. But the question is, is he expected to win 1–1 and recover the ball as a CAM?
- He made a lot more forward and direct passes in the second half, contrasting a larger number of more conservative, backward-looking passes made in the first half. There could be two reasons: (1) there is a general sense of urgency within the Germany team in the second half (2) the introduction of Mario Gómez as a Central Forward really produced an outlet for Özil's key passes, as we see a total of 6 passes directly into the penalty area, three times as many as he did in the first half.

What I found interesting was the heat-pass map of Timo Werner, who started out as the lone striker for the Germany team then paired up with Mario Gomez for much of the second half.





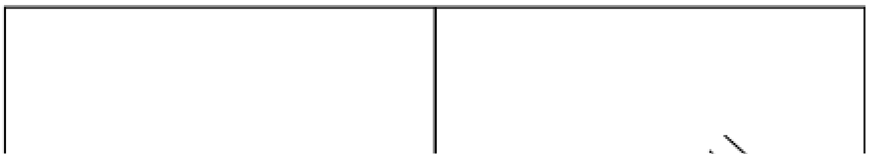
He surprisingly spent a lot of his time on the two sides, while you would expect the Central Forward to occupy the space in the 18-yard box a lot more. This partly explains the ineffectiveness of German offensive line during the game, as their forward lines (Werner, Reus, Goretzka and then Muller, Gómez) crowd up at the wings but fail to take up space in the penalty area, thus providing very little outlet for playmakers such as Özil and Kroos to direct the ball into the 18-yard box.

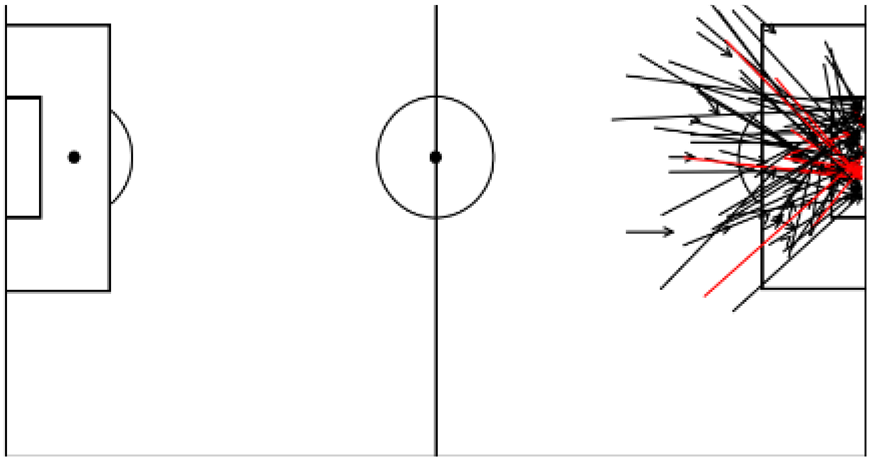
4. Testing your skills: The case of France long-range efforts

A friend of mine was very convinced that the key to France's successful World Cup was their relentless attempts to break down defending lines with long-range efforts. Think about that stunning goal Benjamin Pavard scored against Argentina in the Quarter Final

We can again attempt to visualize all shots from the France team to decide whether the majority of their goals come from outside or inside the box?

If I just follow the methods shown thus far, this is what I get



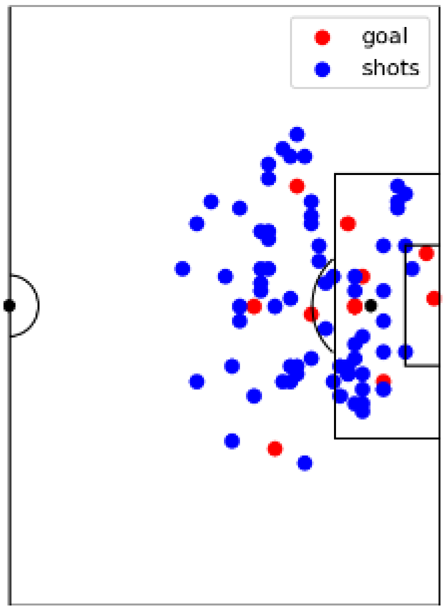


Shot taken by France team during the World Cup campaign

This is fine. But we can do more to make the visualization more engaging and insightful. Specifically, I made two small tweaks:

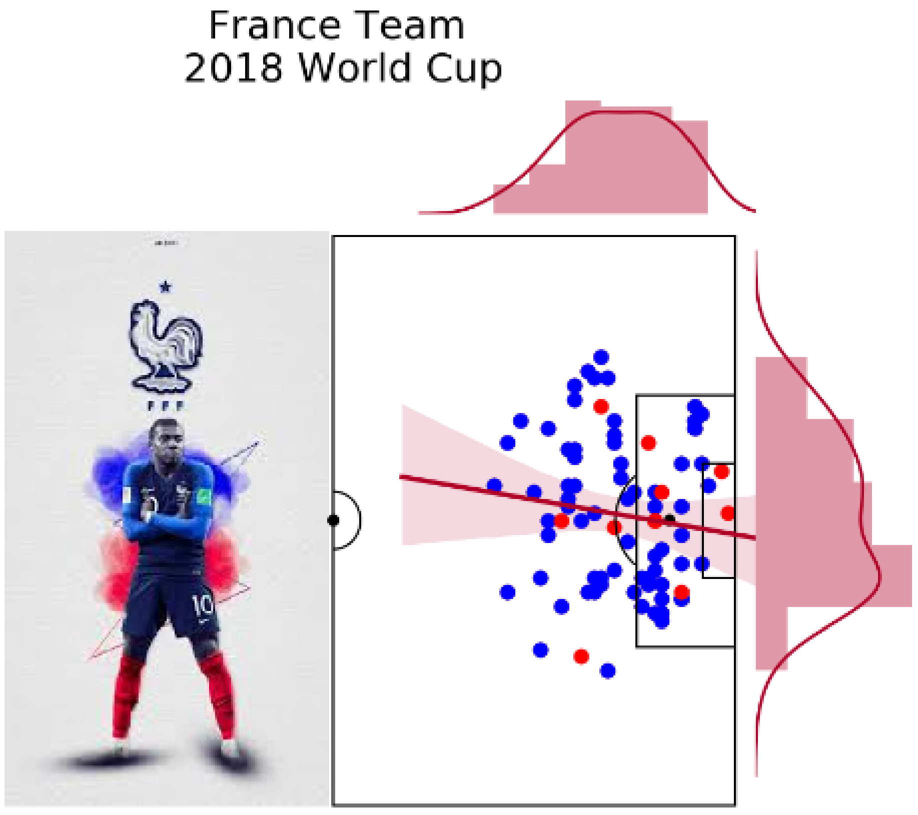
(1) Since we only focus on the shots, which are all recorded at one side of the pitch, I will draw only the right half of the pitch

(2) Since we only care about the starting points of the shots, we can toss away the arrows and only visualize shots as scatter plots where **x, y** are location at which the shot were attempted.



Now this looks a whole lot better. We can see right away that France attempted as many shot inside the boxes as they did outside the penalty area. Although to a certain extent, it does support the argument that France did take a lot more long-range efforts than usual, as we would expect a much lower density of shots outside the box. In any case, it does look interesting how they seems equally clinical with the short and long-range efforts.

In my Jupyter notebook, you can also find further techniques such as overlaying a density plot and including an image to your visualization. With a couple more lines of code, you can easily produce this visualization:



You can find all the source code for this tutorial here

The data used for this project was released by Statsbomb and available to the public here

Get this newsletter Not you?

Soccer Data Science Data Visualization Programming Towards Data Science